Constructing Background Images using Voronoi Tessellation

Version 0.0.1

May 28, 2013

John C. Houck

Kavli Institute for Astrophysics and Space Science,
Massachusetts Institute of Technology, Cambridge MA 02139

1. Motivation

The process of locating sources in *Chandra* observations can be automated using `wavdetect` (Freeman et al. 2002). This method was used to compile a list of X-ray sources observed by *Chandra* for Release 1 of the Chandra Source Catalog (Evans et al. 2010). Automatic source detection in CCD data is complicated by spatial variations in the effective area, especially along chip edges, and by the presence of readout streaks associated with bright point sources. The presence of spatially varying diffuse emission adds to the difficulty. A common problem is that `wavdetect` tends to report large numbers of spurious sources associated with chip edges, readout streaks, and diffuse extended sources. Because a source must, by definition, rise above the local background, a suitable definition of the background at every point in the field of view can can greatly reduce the number of spurious sources reported by `wavdetect`. Here, I describe a method for constructing a background image for input to `wavdetect` that reduces the number of spurious sources associated with chip edges, readout streaks and diffuse extended sources.

2. Method

Note that the Voronoi tessellation approach used here is based on the ideas used in `vtpdetect` (Ebeling & Wiedenmann 1993).

2.1. Voronoi Tessellation of a Poisson Point Process

One can idealize the distribution of background counts as a homogeneous Poisson point process that deposits N events at random points with uniform spatial distribution over the detector area, Ω, within time, t. Given such a set of points, the Voronoi tessellation determines a unique set of non-overlapping polygons, each containing a single event, such that the set of polygons covers the entire detector region Ω. It is useful to define a dimensionless reduced polygon area as $\alpha = a/\bar{a}$, where $\bar{a}$ is a suitably chosen value that will be described below.

The statistical distribution of Poisson Voronoi cells has a form that is well characterized by simple analytic functions (Tanemura 2003). In particular, the distribution of Voronoi polygon reduced areas, α, converges to a generalized gamma distribution of the form

$$f(\alpha) = \frac{ab^{c/a}}{c(c/a)} \alpha^{c-1} \exp\left(-b\alpha^{a}\right), \tag{1}$$

where the parameters $\{a, b, c\}$ have been empirically found to be $\{1.07950, 3.03226, 3.31122\}$ (Tanemura 2003).

When sources are present, the spatial distribution of events is nonuniform (see Figure 1). In particular, the events near a source are more closely spaced than the background events, making the source polygons smaller on average than the background polygons. To detect the sources in an image, one can construct the Voronoi tessellation and then associate with sources all polygons having reduced area $\alpha < \alpha_{\text{thresh}}$, where α_{thresh} is a suitably defined reduced area threshold (Ebeling & Wiedenmann 1993).

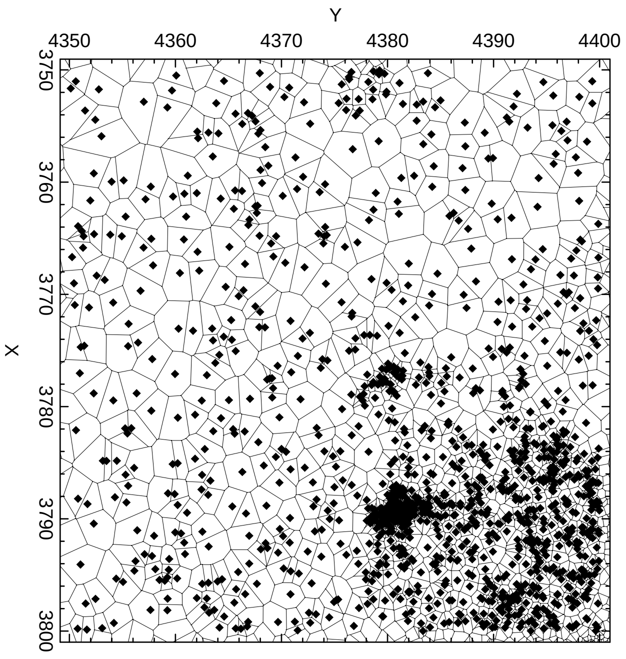

Fig. 1.— Voronoi tessellation of a region containing both source and background events.

2.2. Polygon Area Threshold

Using the sky X, Y coordinates of events in a given observation it is straightforward to construct the Voronoi tessellation. The implementation described here constructs the Voronoi tessellation using the `Triangle` code developed by Shewchuk (1996).

It is convenient to process the events on each CCD separately. To account for spatial variations in the effective area, the geometric area of each Voronoi polygon, $a_{\mathrm{geom},k}$, can be corrected using the normalized effective area, $A_k \in [0,1]$, determined from the exposure map (Davis 2001; Ebeling & Wiedenmann 1993). A histogram of corrected polygon areas, $a_k = a_{\mathrm{geom},k} A_k$, shows the actual distribution of polygon areas in a given observation. For the largest polygon areas, the shape of this empirical distribution is expected to correspond closely to the shape defined by equation (1), given an appropriate value for the mean polygon area, $\bar{a}$.

An appropriate mean polygon area, $\bar{a}$, can be determined iteratively by fitting the large area tail of the observed histogram of polygon areas with a function of the form

$$F_i(a; \mathcal{N}, \bar{a}) = \mathcal{N} \int_{a_i}^{a_{i+1}} f(a/\bar{a}) \mathrm{d}a \tag{2}$$

where the ith histogram bin spans the interval $[a_i, a_{i+1})$ and where the overall normalization, $\mathcal{N}$, depends on total number of events in the observation. The integral, F_i, is related to the normalized incomplete Gamma function and may be evaluated conveniently using the GSL library function `gsl_sf_gamma_inc_P` (Galassi et al. 2005).

As an initial guess, one can use $\bar{a}_1 = \sum a_k / N$, restricting the fit to histogram bins in the range $[\bar{a}_1, \max(a_k)]$. Fitting equation (2) to the observed histogram with $\bar{a}$ as an adjustable parameter yields a new value for the mean area, $\bar{a}_2$. Restricting the fit to the range $[\bar{a}_2, \max(a_k)]$ and fitting again yields an improved value, $\bar{a}_3$. After a few iterations, this process should converge to a good estimate of the mean polygon area, $\bar{a}$, consistent with the background on the chip.

Voronoi source polygons can now be identified as those polygons with area smaller than

$$a_{\mathrm{thresh}} = \alpha_{\mathrm{thresh}} \bar{a} \tag{3}$$

where α_{thresh} is a user adjustable parameter; typically, $\alpha_{\mathrm{thresh}} \approx 0.25$.

Adjacent Voronoi source polygons can be merged efficiently using the polygon merge algorithm described by Zalik (2003). Note that the resulting merged polygons are no longer convex and may contain holes (see Figure 2).

After constructing a set of merged source polygons, one should examine the fractional area of the CCD covered by these polygons. If most of the area of the CCD is covered by source polygons, then either the value of $\bar{a}$ is too large, or the observation contains too little background to be processed effectively by this algorithm. Reducing the value of $\bar{a}$ and re-defining the source polygons may greatly reduce the total area covered by source polygons. If not, then processing should stop at this point.

Once a value of $\bar{a}$ is selected and a set of merged Voronoi source polygons is constructed, the

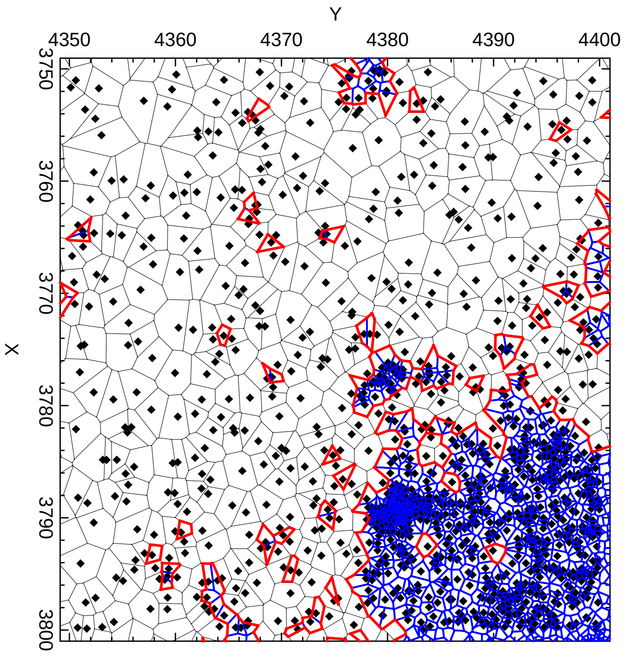

Fig. 2.— Voronoi tessellation of a region containing both source and background events. Voronoi polygons with areas $a < \alpha_{\rm thresh}$ (blue) are identified with sources. Merging Voronoi source polygons yields non-convex source polygons (red).

above-background regions of the image are taken to be those merged polygons that contain at least $\mathcal{N}_{\rm s}$ counts. Typically, $\mathcal{N}_{\rm s} \gtrsim 3$.

2.3. Readout Streaks

Given a set of merged source polygons, P_j, one can detect readout streaks in the following way.

1. Rotate the set of merged source polygons into an x', y' coordinate system in which the readout direction lies along the y' direction.

2. Generate a large number of uniformly distributed x', y' points.

3. For each randomly generated point that falls inside a source polygon, record which polygon it fell in and increment a counter for the corresponding x' column.

The number of counts in each x' column is proportional to the fraction of that column's area that falls within source polygons. Those columns that are mostly (e.g $\sim$80%) covered by source polygons are labeled as readout streaks and the associated source polygons are labeled as streak polygons.

After identifying streak polygons, one should consider the fraction of CCD columns flagged as readout streaks. If too many columns have been flagged, it may be that the value of $\bar{a}$ is too large. Reducing the value of $\bar{a}$ and re-defining the source polygons may greatly reduce the total area covered by source polygons. If not, then processing should stop at this point.

2.4. Filtering Source Events

Given a set of merged source polygons, P_j, the events in the dataset can be partitioned into two subsets, a set of source events, $S \in \{P_j\}$ and a set of background events, $B \notin \{P_j\}$. To construct a background image we must filter the set S.

To filter the events contained in the set of source polygons, S, we may further partition the S polygons into three categories: *small* polygons ($a \leq a_{\rm large}$), *large* polygons ($a > a_{\rm large}$), and *streak* polygons. The area threshold $a_{\rm large}$ is chosen to distinguish point sources from extended sources such as galaxies.

Events falling in small polygons are replaced with a smaller number of uniformly distributed points. The number of replacement points is chosen to match the spatial density of background counts, $n_k = f_{\rm small} a_k / \bar{a}$, where $f_{\rm small}$ is a control parameter.

Events falling in large polygons may also be replaced with a number of uniformly distributed points. Because such extended sources may contain bright point sources (e.g. a galaxy with a bright AGN), the spatial density of replacement points may be chosen high enough to mask the diffuse emission, allowing `wavdetect` to pick out the bright embedded point sources. The intent is to reduce the number of spurious point sources associated with diffuse emission from extended sources.

When generating coordinates for spatially uniformly distributed points, we use low discrepancy Halton sequences to reduce the chance of creating a local grouping of randomly generated points that could be mistaken for a faint source.

To reduce the number of spurious sources associated with readout streaks, we want to include all of the streak events in the background image. But because the streak polygons may also contain embedded bright point sources, some filtering of streak polygon events is useful. To filter events in streak polygons, we construct an image of the events, select those pixels exceeding a specified threshold, and filter out enough events in those pixels to bring them below the specified threshold.

2.5. Background Image Construction

After filtering events contained in source polygons of all types, a background image is constructed from the remaining events using the Delaunay Triangulation Field Estimator (DTFE) described by Schaap (2007). We use `Triangle` (Shewchuk 1996) to construct the Delaunay triangulation.

Given the Delaunay triangulation of a point set, the DTFE provides an estimate of the count density at any spatial point. This estimate is based on a 2D linear interpolation between count density values defined at each vertex in the triangulation. To construct the estimate, we must first construct an appropriate definition of the count density, σ_i, at each vertex in the triangulation.

The Delaunay triangulation defines a set of M triangles such that the i^{th} triangle has area $\hat{A}(i)$. Using a linear approximation to the count density within each triangle, the total number of counts in each triangle corresponds to the volume of the polyhedron shown in Figure 3. Note that the resulting number of counts need not be an integer.

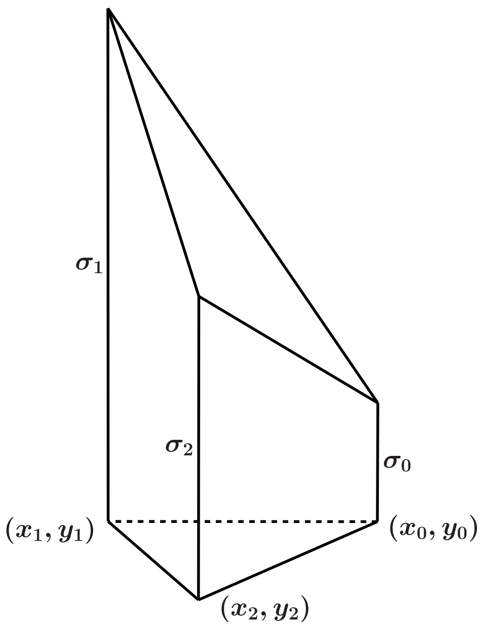

Fig. 3.— Polyhedron volume element.

The volume of the polyhedron shown in Figure 3 is

$$\hat{V} = \frac{1}{3}\left(\sigma_0 + \sigma_1 + \sigma_2\right)\hat{A}, \tag{4}$$

where $\hat{A}$ is the area of the triangle with vertices $(x_j, y_j), j = 1, 2, 3$. Expressing the total number of counts as a sum of such polyhedron volumes, we can write

$$N = \sum_{i=1}^{M} \hat{V}(i) = \sum_{i=1}^{M} \frac{1}{3}\left[\sigma_0(i) + \sigma_1(i) + \sigma_2(i)\right]\hat{A}(i). \tag{5}$$

Because each vertex, k, in the triangulation is shared by a number of triangles, M_k, each σ_k will appear M_k times in the sum in equation (5). Rearranging the terms in that sum, we can collect the coefficients of each σ_k together, writing the rearranged expression in the form

$$N = \frac{1}{3}\sum_{k=1}^{N}\sigma_k \sum_{j=1}^{M_k}\hat{A}_k(j), \tag{6}$$

where we now have a sum over N vertices. The notation $\hat{A}_k(j)$ means that triangle j containing vertex k has area $\hat{A}_k(j)$. Equation (6) is trivially satisfied if we define

$$\sigma_k \equiv \frac{3}{\sum_{j=1}^{M_k}\hat{A}_j(k)}, \tag{7}$$

as the count density associated with each vertex.

Having defined the count density at each point in the triangulation, it is straightforward to construct the DTFE image by simply integrating over the continuous (linearly interpolated) count density function to determine the number of counts in every image pixel. This integration can be directly expressed as a sum of polyhedron volumes in which every polyhedron has the form shown in Figure 3.

The DTFE image has the advantage that it preserves exactly the total counts in the data and linearly interpolates the spatial structure into every pixel while preserving small scale features associated with bad columns, chip edges, and readout streaks (see Figure 4).

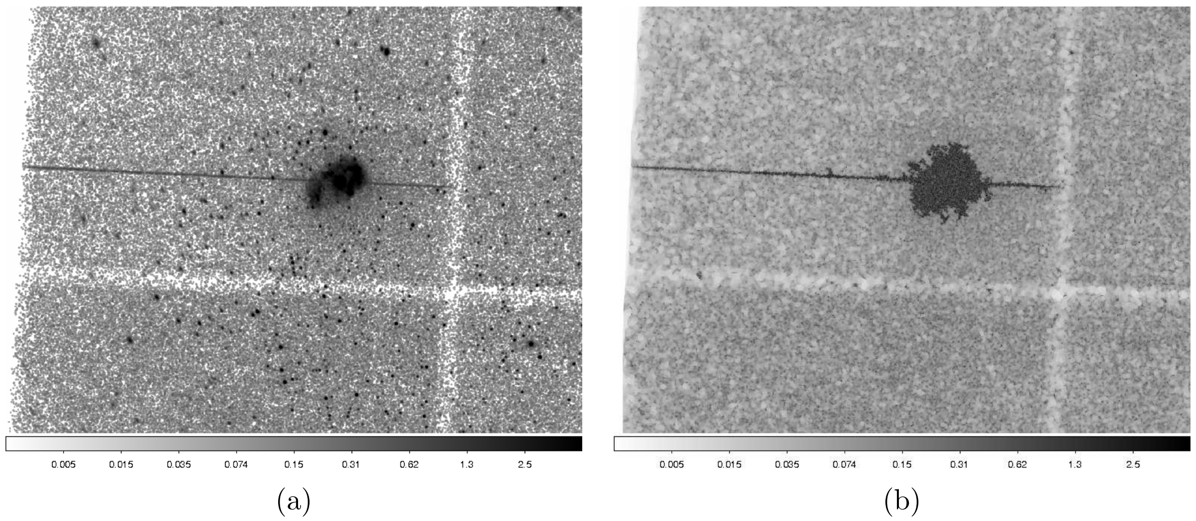

Fig. 4.— (a) *Chandra* level 2 event file for `obsid` 6204, smoothed with a Gaussian kernel with radius 3 pixels. (b) DTFE background image for the same region constructed using the algorithm described. Note that the background image contains the readout streak and that the polygon containing the galaxy has been filled in with a relatively high uniform surface brightness. Both images are grey scale, showing pixel values in the range 0.001 (white) to 5 counts (black) on a logarithmic scale.

2.6. Issues

Using the DTFE background image in `wavdetect` reduces the number of spurious sources associated with chip edges and bright readout streaks (see Figure 5).

However, in each observation, a few sky X, Y pixels not obviously associated with any source may contain multiple events, yet still fall below the source count threshold, $\mathcal{N}_{\rm s}$. Such pixels may be associated with unfiltered afterglow events or may occur when post-processing with a sub-pixel algorithm leaves multiple background events with identical X, Y coordinates. For obvious reasons, the tessellation algorithm described here requires every event to have unique X, Y coordinates. To ensure that that is the case, we apply a tiny random offset, e.g. 0.1 pixel to all input events. Nevertheless, in each observation it may happen that up to $\mathcal{N}_{\rm s}$ "background" events may occur in a few single, isolated ACIS pixels in sky X, Y coordinates, Whenever this happens, the corresponding pixels in the DTFE background image will be very bright compared to the typical nearby background level.

While such bright pixels may appear incorrect, they do accurately reflect the estimated background count distribution. It is unclear how to avoid these bright pixels without breaking the mechanism used to avoid spurious sources along readout streaks, or otherwise limiting the effectiveness of the overall method.

REFERENCES

Davis, J. E., 2001, ApJ, 548, 1010

Ebeling, H., & Wiedenmann, G., 1993, Phys. Rev. E, 47, 704

Evans, I. N., et al., 2010, ApJS, 189, 37

Freeman, P. E., Kashyap, V., Rosner, R., & Lamb, D. Q., 2002, ApJS, 138, 185

Galassi, M., Davies, J., Theiler, J., Gough, B., Jungman, B., Booth, M., & Rossi, F., 2005, GNU Scientific Library Reference Manual - Revised Second Edition, (Bristol: Network Theory Ltd)

Schaap, W., 2007, *Ph.D. thesis*, Rijksuniversiteit Groningen

Shewchuk, J. R., 1996, in Applied Computational Geometry: Towards Geometric Engineering, ed. M. C. Lin, D. Manocha, Vol. 1148, Lecture Notes in Computer Science, Springer-Verlag), 203–222, From the First ACM Workshop on Applied Computational Geometry

Tanemura, M., 2003, Forma, 18, 221

Zalik, B., 2003, Engineering with Computers, 19, 35, 10.1007/s00366-002-0247-6

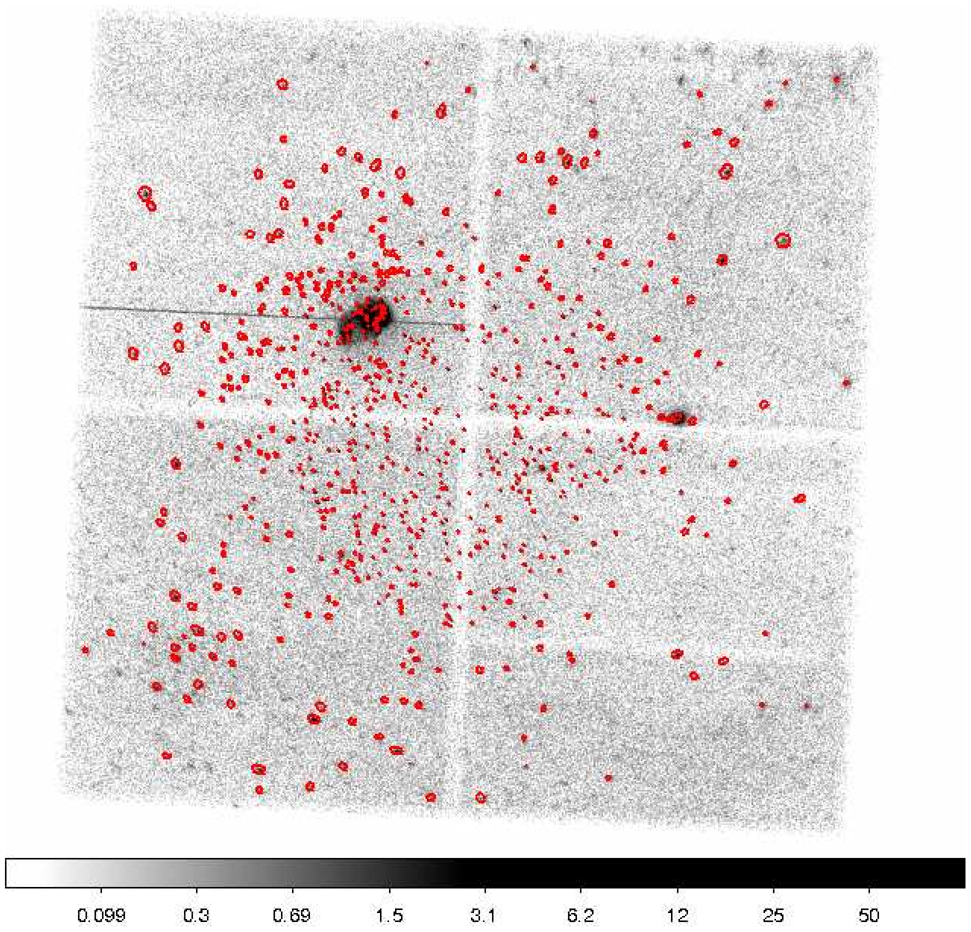

Fig. 5.— The red ellipses show the X-ray sources found by `wavdetect` using the DTFE background image shown for *Chandra* `obsid` 6204 in Figure 4. Note that there are no spurious sources associated with the readout streak; the few sources detected near the streak are undoubtedly real sources.